M11102122 陳煜翔 影像處理 Homework 2

I package all function which I will use in the cv_image class.
The object have image and new_image for original image and the histogram equalization image.

Function of convert2gray is used to conver BGR image to Grayscale.

Function of gamma_correction_image is use gamma correction function to make image lighter of darker. In our task we use for darker.

Function of plt_histogram is used to get the histogram information and save the figure of it.

Function of Histogram_equalization is used to calculate the new image for algorithm.

utils.py

C: > Users > Skyler > Desktop > Course > Image_Processing > LAB2 > utils.py > cv_image > gamma_correction_image

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

class cv_image:
    def __init__(self, Image):
        self.image = cv2.resize(Image, (1024, 1024))
        self.new_image = cv2.resize(Image, (1024, 1024))

    def convert2gray(self): # Convert Image from BGR to GRAY
        self.new_image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)

    def gamma_correction_image(self, gamma): # Use gamma correction to change illumination of image
        inverse_gamma = 1.0 / gamma
        self.new_image = self.image.astype(np.float32) / 255.0 # Normalize image to [0, 1]
        self.new_image = ((self.new_image ** inverse_gamma) * 255.0 ).astype(np.uint8)

    def plt_histogram(self, img, name, save): # Get histogram value
        hist = plt.hist(img.flatten(), 256, [0, 255], label = name)
        plt.xlabel("Value")
        plt.ylabel("Frequency")

        if save == True:
            plt.legend([name], loc ="lower right")
            plt.savefig(name + ".png")
            plt.close("all")

        return hist

    def histogram_equalization(self, q_k, q_o, hist):
        width, heigh = self.new_image.shape # Save the orignal shape for the resize to same size.

        img = self.new_image.flatten() # Convert to 1-D vector
        cp_img = img.copy()

        fq_sigma = 0
        for i in range(img.size):
            fq_sigma = hist[:cp_img[i]+1].sum() # Calculate sigma(H(p))
            img[i] = (((q_k - q_o) / img.size) * fq_sigma) + q_o # histogram_equalization function

        img = np.resize(img, (width, heigh)) #  Resize to the original size
        return img
```

	
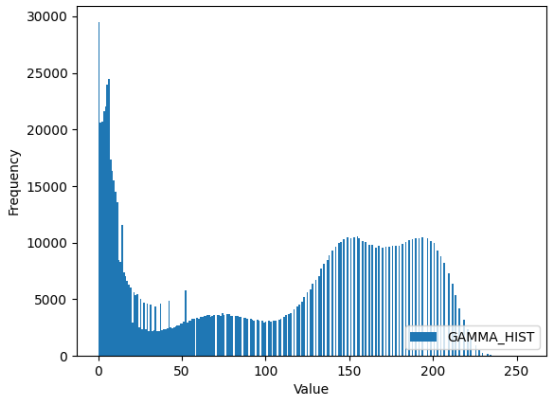	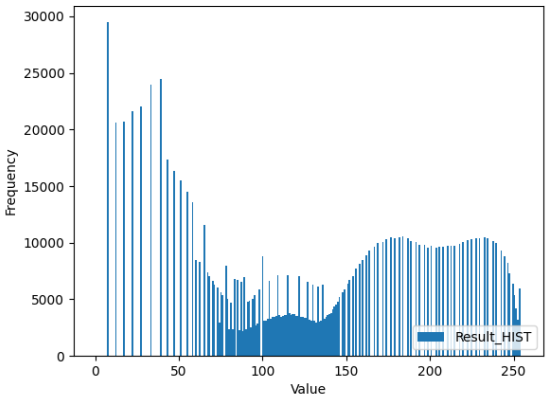
Gamma Image (Dark image)	Result (Histogram Equalization)

Image Processing Homework 2

Author: M11102122 陳煜翔

Date: 2022/10/21

[TOC]

Introduction

In this task, we use gamma correction to darken the image, and then achieve histogram equalization to get the image.

Coding Detail

Environment

We use Python environment by version **3.8**, and use cv2, numpy to achieve the histogram equalization algorithm.

Constant

The constant value is setting in the config.py file.

Image_Path: The original picture path. q_k: Maximum choose gray scale value. q_o: Minimum Choose gray scale value.

```
# config.py
Image_Path = r".\0.jpg"
q_k = 255
q_o = 0
```

```
# main.py
import utils
import config
import cv2

if __name__ == "__main__":
    Image_Path = config.Image_Path
    Imag = cv2.imread(Image_Path)
    image = utils.cv_image(Imag)
    image.convert2gray()
    image.gamma_correction_image(gamma=0.4)

    raw_hist = image.plt_histogram(img=image.image, name="RAW_HIST", save=True)
    gamma_hist = image.plt_histogram(img=image.new_image, name="GAMMA_HIST",
save=True)
```

```
    q_k = config.q_k
    q_o = config.q_o
    img = image.histogram_equalization(q_o=q_o, q_k=q_k, hist=gamma_hist[0])

    result_hist = image.plt_histogram(img=img, name="Result_HIST", save=True)

    cv2.imwrite("Result.png", img)
    cv2.imwrite("Gamma_image.png", image.new_image)
```

```
# utils.py
import cv2
import numpy as np
import matplotlib.pyplot as plt

class cv_image:
    def __init__(self, Image):
        self.image = cv2.resize(Image, (1024, 1024))
        self.new_image = cv2.resize(Image, (1024, 1024))

    def convert2gray(self): # Convert Image from BGR to GRAY
        self.new_image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)
        self.image = cv2.cvtColor(self.image, cv2.COLOR_BGR2GRAY)

    def gamma_correction_image(self, gamma): # Use gamma correction to change
illumination of image
        inverse_gamma = 1.0 / gamma
        self.new_image = self.image.astype(np.float32) / 255.0 # Normalize image
to [0, 1]
        self.new_image = ((self.new_image ** inverse_gamma) * 255.0
).astype(np.uint8)

    def plt_histogram(self, img, name, save): # Get histogram value
        hist = plt.hist(img.flatten(), 256, [0, 255], label = name)
        plt.xlabel("Value")
        plt.ylabel("Frequency")

        if save == True:
            plt.legend([name], loc ="lower right")
            plt.savefig(name + ".png")
            plt.close("all")

        return hist

    def histogram_equalization(self, q_k, q_o, hist):
        width, heigh = self.new_image.shape # Save the orignal shape for the
resize to same size.

        img = self.new_image.flatten() # Convert to 1-D vector
        cp_img = img.copy()

        fq_sigma = 0
```

```
        for i in range(img.size):
            fq_sigma = hist[:cp_img[i]+1].sum() # Calculate sigma(H(p))
            img[i] = (((q_k - q_o) / img.size) * fq_sigma) + q_o #
histogram_equalization function

        img = np.resize(img, (width, heigh)) #  Resize to the original size
        return img
```

Design Detail

We use cv2 package to read Image, and resize it to 1024x1024. Futher we use gamma correction to get the darken image for the experiment.

Then we use the darken image to achieve the histogram equalization.

Experiment Result